PART 6 – KNOWLEDGE BASED SYSTEMS

Njeri Ireri

July – October 2021

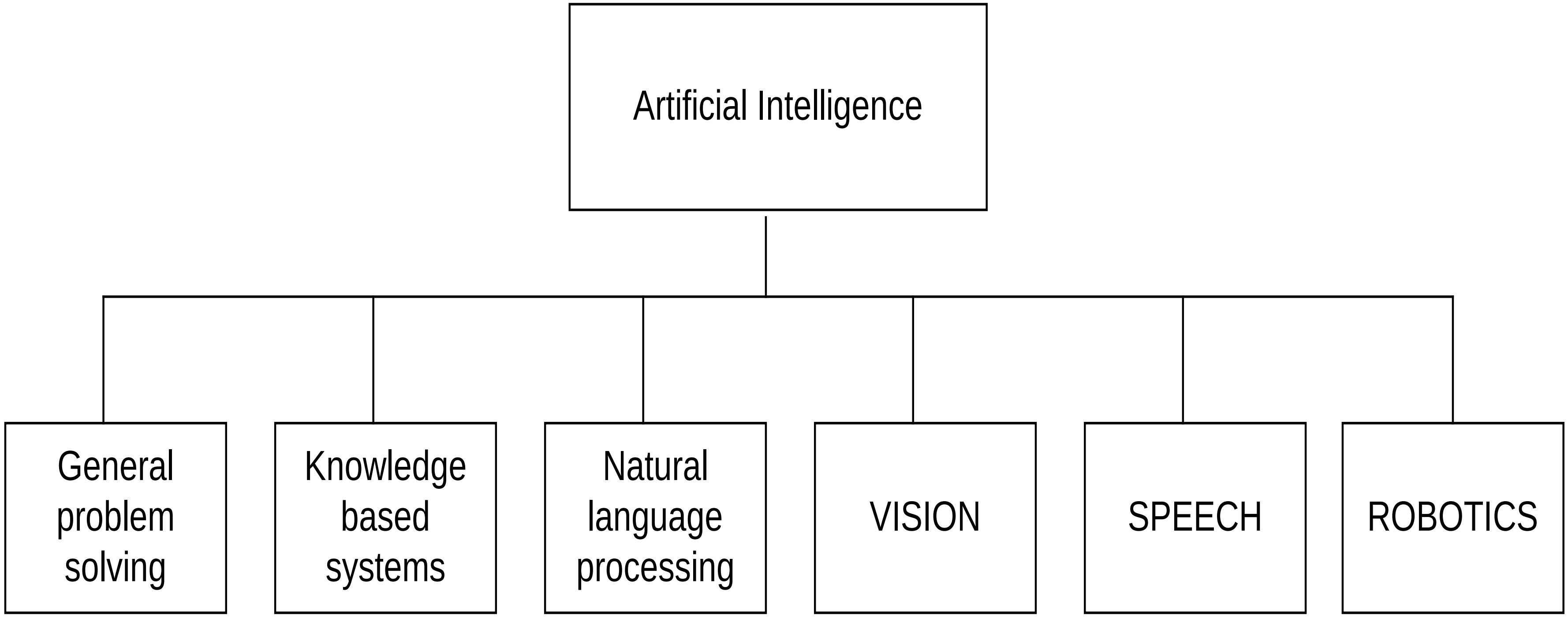
Introduction to KBS

Overview

- AI and KBS
- Definition, Components,
- Application areas

Introduction to KBS

- Artificial Intelligence and Knowledge Based Systems

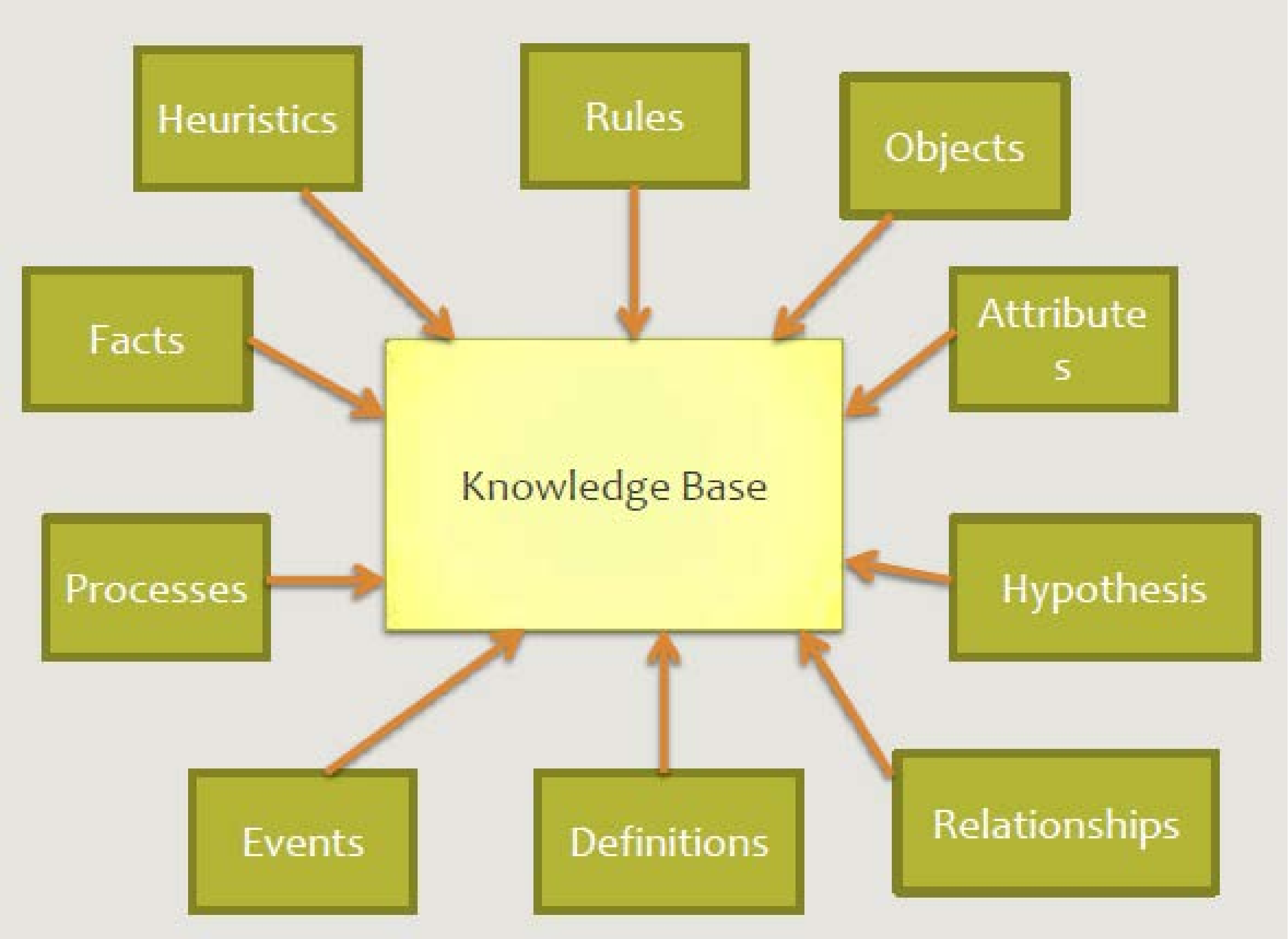


Introduction to KBS

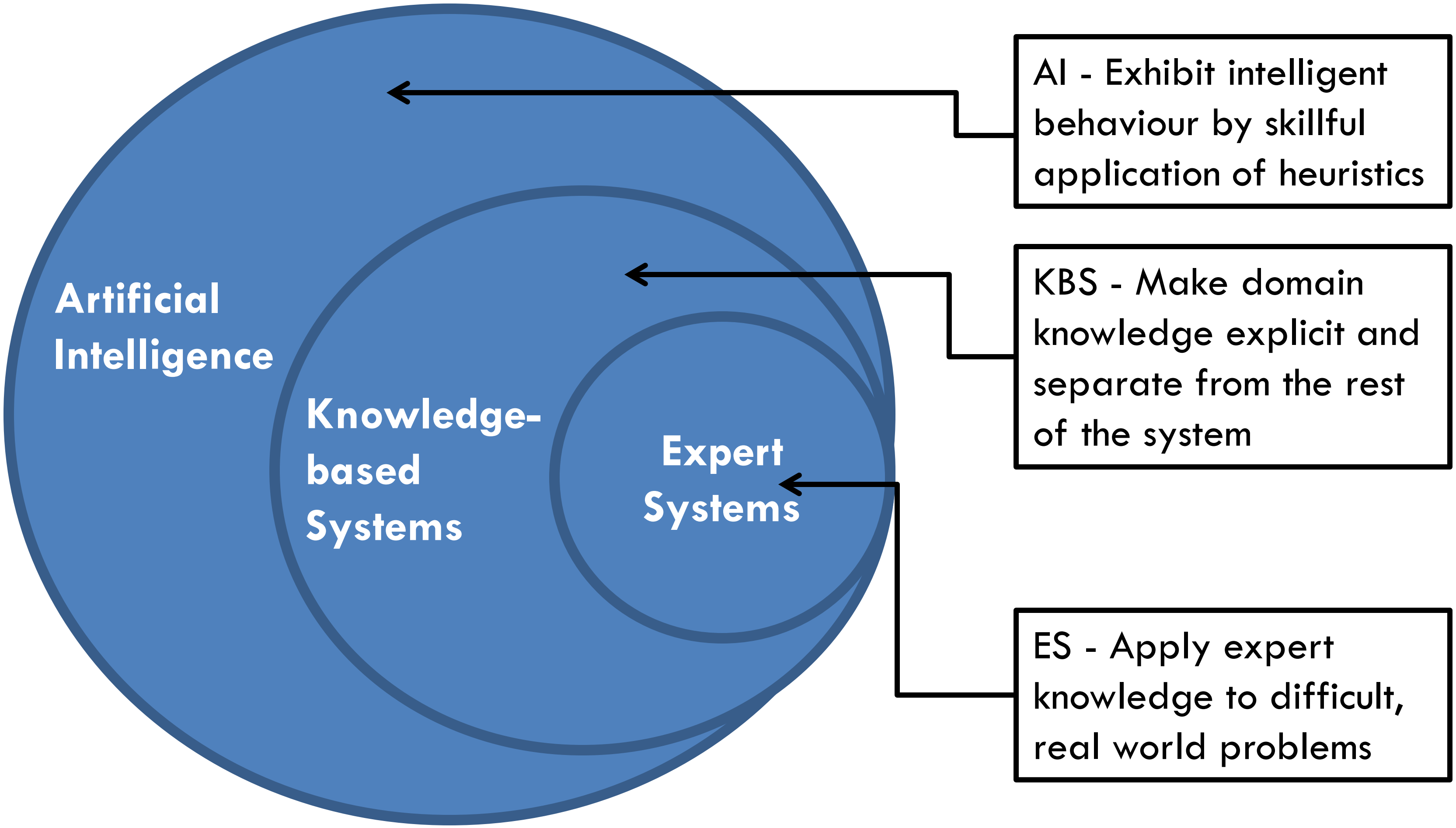
- KBS definition
  - A Knowledge Based Systems is an AI. System that models human expertise in a limited area
  - A system which is built around a knowledge base. i.e. a collection of knowledge, taken from a human, and stored in such a way that the system can *reason* with it
  - A computer system whose usefulness derives primarily from a data base containing *human knowledge in a computerized format*

- Knowledge-based systems are also known as advisory systems, knowledge systems, intelligent job aid systems, or operational systems

Introduction to KBS



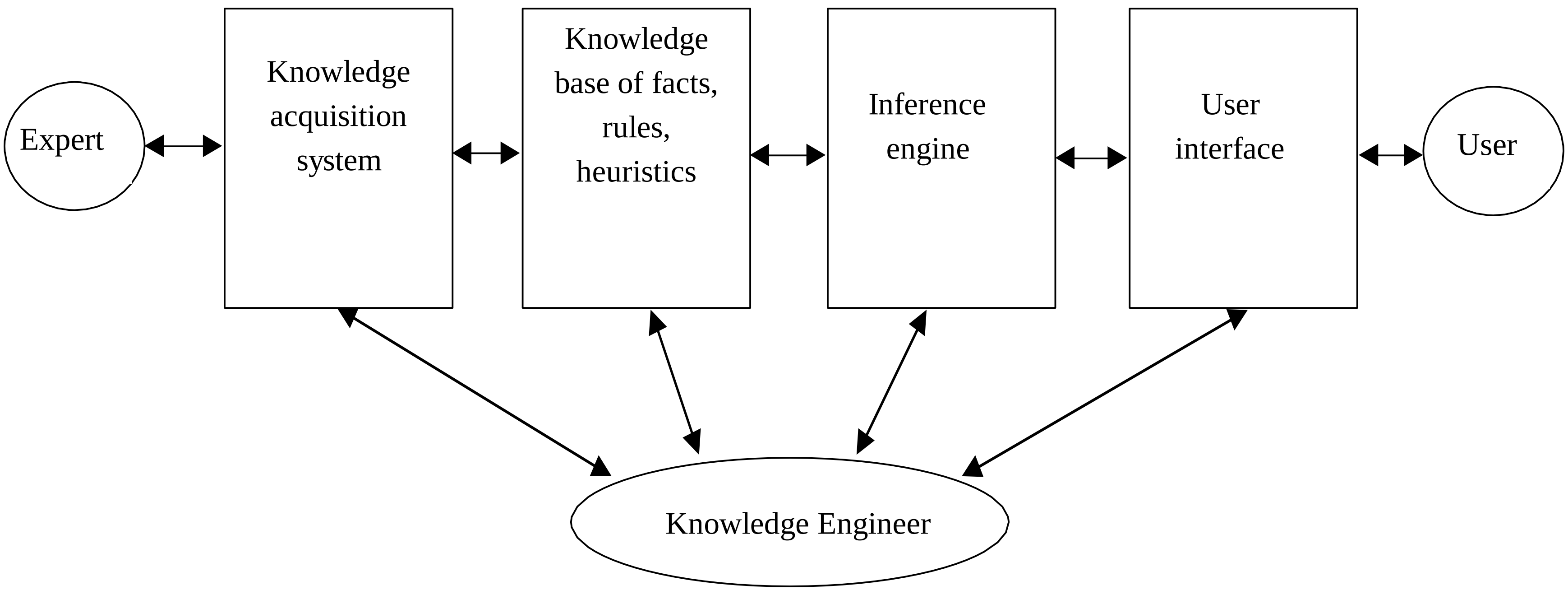
AI, KBS and Expert Systems



Introduction to KBS

- Expert System definition
  - "a computer program that represents and reasons with knowledge of some specialist subject with a view to solving problems or giving advice." – Peter Jackson
  - "an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solutions." – Edward Feigenbaum
  - "a system that users human knowledge captured in a computer to solve problems that ordinarily require human expertise." – Efraim Turban and Jay Aronson

Components and Structure of a KBS



Building expert systems is generally an iterative process - The components and their interaction will be refined over the course of numerous meetings of the knowledge engineer with the experts and users

Creating an KBS

- Two steps involved:
  1. extracting knowledge and methods from the expert (knowledge acquisition)
  2. reforming knowledge/methods into an organised form (knowledge representation)

KBS Application Areas

- Generally, KBS have been found useful for certain sorts of problems. Here are some examples:
  - Interpretation - Inferring situation descriptions from observations OR inferring situation descriptions from sensor data
  - Prediction - Inferring likely consequences of given situations.
  - Diagnosis - Inferring system malfunctions from observations.
  - Design - Configuring objects under constraints.
  - Planning - Developing plans/designing actions.
  - Monitoring - Comparing objects under construction.
  - Debugging - Prescribing remedies for malfunctions.

KBS Application Areas

- Repair - Executing a plan to administer a prescribed remedy.
- Instruction - Diagnosing, debugging, and correcting student performance.
- Advice - providing suitable option(s) under some circumstances
- Control - governing overall system behavior
- Prescription
- Scheduling
- Selection
- Taxonomy

Knowledge Acquisition

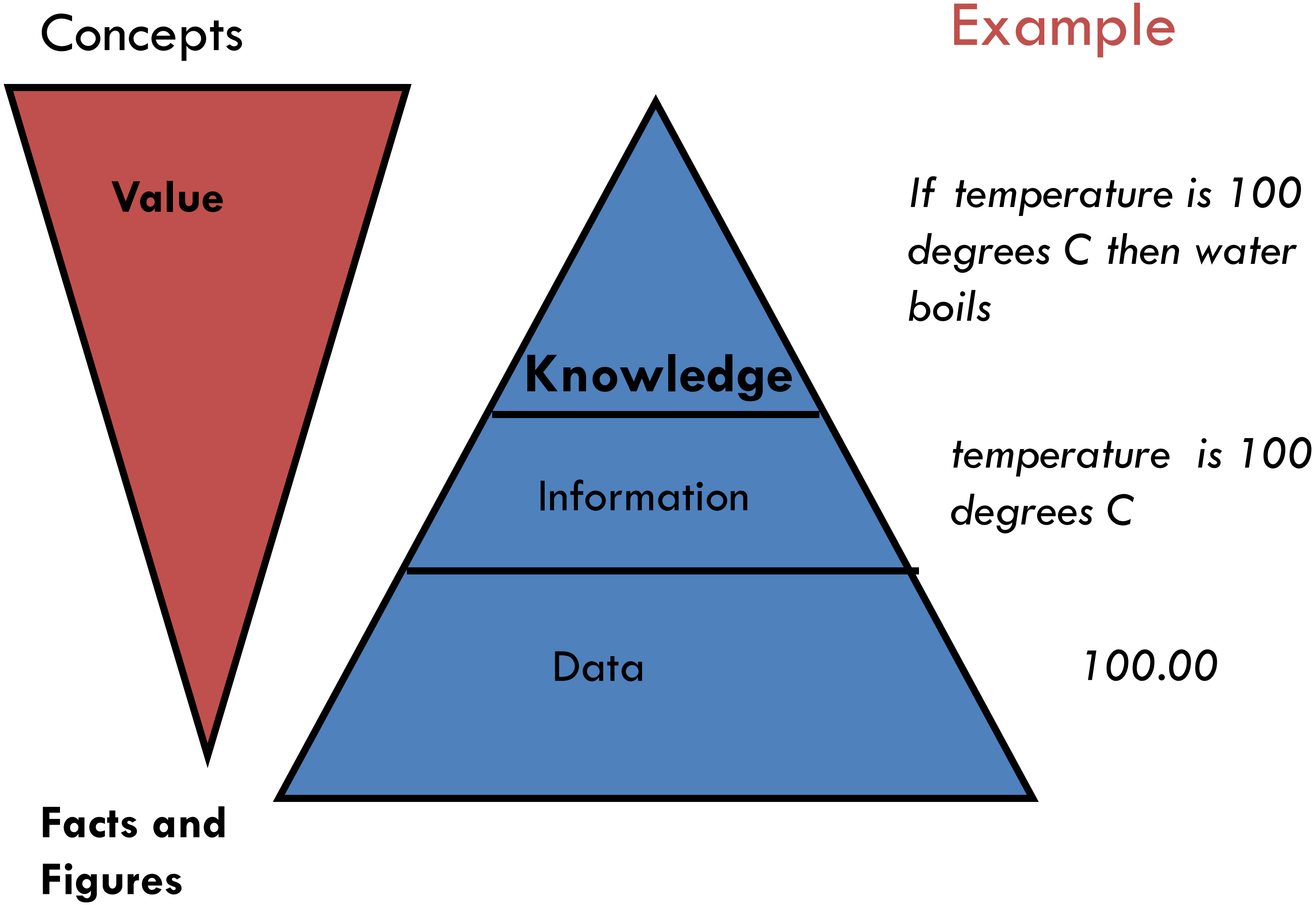
We Shall Discuss

- What is Knowledge?
- Sources of Knowledge
- Levels of knowledge
- Categories of knowledge
- What is Knowledge Acquisition?
- Stages of Knowledge Acquisition
- Role of the Knowledge Engineer
- Methods of Knowledge Acquisition
- Knowledge Acquisition Difficulties
- Organizing the Knowledge

Data, Information and Knowledge

- What is knowledge?
  - **Data:**
    - Raw facts, figures, measurements
  - **Information:**
    - Refinement and use of data to answer specific question
  - **Knowledge:**
    - Refined information
    - True rational belief(philosophy).OR facts, data and their relationships (Computational view)
    - Knowledge is the sort of information that people use to solve problems.
    - Knowledge includes: facts, concepts, procedures, models, heuristics, examples.

Data, Information and Knowledge



Sources of Knowledge

- **Documented**
  - books, journals, procedures
  - films, databases
- **Undocumented**
  - people's knowledge and expertise
  - people's minds, other senses
  - This is mainly **tacit/implicit** knowledge (e.g. procedural knowledge) that is difficult to capture
    - It is implied by or inferred from actions or statements of experts

Knowledge Acquisition

- **Knowledge acquisition** is the process of extracting, structuring and organizing **knowledge** from one source, usually human experts, so it can be used in Knowledge based systems
- This is often the major obstacle in building a KBS Systems
  - Knowledge acquisition is a multifaceted problem that encompasses many of the technical problems of knowledge engineering, the enterprise of building knowledge base systems (Gruber)

Knowledge Acquisition

- The knowledge acquisition process is usually comprised of three principal stages:
  - **Knowledge elicitation** is the interaction between the expert and the knowledge engineer/program to elicit the expert knowledge in some systematic way
  - The knowledge thus obtained is usually stored in some form of human friendly **intermediate representation**
  - The intermediate representation of the knowledge is then compiled into an **executable form** (e.g. production rules) that the inference engine can process

Context - Developing Expert Systems

- Who is involved?
  - **Knowledge Engineer**
    - A knowledge engineer is a computer scientist who knows how to design and implement programs that incorporate artificial intelligence techniques
    - Interacts between expert and Knowledge Base
    - Needs to be skilled in extracting knowledge
    - Uses a variety of techniques
  - **Domain Expert**
    - A domain expert is an individual who has significant expertise in the domain of the expert system being developed

Methods of Knowledge Acquisition

- Manual:
  - interview with experts
    - structured, semi structured, unstructured interviews
  - track reasoning process and observing
- Semi Automatic:
  - use a computerised system to support and help experts and knowledge engineers
    - Repertory Grid Analysis
- Automatic:
  - minimise the need for a knowledge engineer or expert
  - Broadly, this means using a computer program to convert data into knowledge. This process may also be described as **learning**
    - Induction Decision Trees (ID3

Knowledge Acquisition Difficulties

- Transferring knowledge from one person to another is difficult. Even more difficult in A.I For these reasons:
  - expressing knowledge
  - The problems associated with transferring the knowledge to the form required by the knowledge base
  - experts busy or unwilling to part with knowledge
  - methods for eliciting knowledge not refined
  - collection should involve several sources not just one
  - it is often difficult to recognise the relevant parts of the expert's knowledge
  - experts change

Knowledge Representation

Overview

- Definition
- Representation schemes/formalisms

Definition

- **Knowledge.** True rational belief(philosophy).*OR* facts, data and relationships (Computational view).

- **Representation.** Structure + operations; *OR* map + operations; *OR* game layout and rules of play; *OR* abstract data types.

- **Knowledge representation.** Framework for storing knowledge and manipulating knowledge *OR* *'Set of syntactic and semantic conventions that makes it possible to describe things.' Bench-Capon, 1990.*

Knowledge-based Agent

- Central component of a knowledge-based agent is its knowledge-base (KB)
- A KB is a set of representations of facts about the world
- Each individual representation is called a sentence
- The sentences are expressed in a language called a knowledge representation language
- A knowledge-based agent should be able to infer

Knowledge Representation

- **The object of Knowledge Representation is to express knowledge in a computer-tractable form, so that it can be used to help agents perform well.**
- A Knowledge Representation language is defined by two aspects:
  - **Syntax**: describes how to make sentences OR describes the possible configurations that can constitute sentences.
  - **Semantics**: determine the facts in the world to which the sentences refer OR the "things" in the sentence.

Knowledge Representation Schemes

- **Different Knowledge Representation schemes/formalisms**
  - Natural Language
  - Rules
  - Logic
    - Propositional logic (Boolean Logic)
    - Predicate logic (First Order Logic)
  - Semantic Nets
  - Frames

Natural Language

Expressiveness of natural language:

- Very expressive, probably everything that can be expressed symbolically can be expressed in natural language (pictures, content of art, emotions are often hard to express)
- Probably the most expressive knowledge representation formalism we have. Reasoning is very complex, hard to model

Problems with natural language:

- Natural language is often ambiguous
- The syntax and semantics are not fully understood
- There is little uniformity in the structure of sentences

Natural Language is Ambiguous

Examples:

- Mary lost a jewel.
  - *Jewel is noun or a verb?*
- Lung cancer in women mushrooms
  - *Mushrooms is noun or a verb?*
- Iraqi Head Seeks Arms
  - *Arms can mean different things, which is it?*
- Two Soviet Ships Collide, One Dies
  - *What does one refer to in this case?*
- Chef throws his heart into feeding needy
  - *Throws his heart is not decomposed normally in this case: idiom*

Rules

- These are formalizations often used to specify recommendations, give directives or strategy.
- ***Format:*** **IF <premises> THEN <conclusion>.**
- Related ideas: rules and fact base; conflict set - source of rules; conflict resolution- deciding on rules to apply.

  - *Advantages:* easy to use; explanations are possible; capture heuristics; can handle uncertainties to some extent.
  - *Disadvantages:* cannot cope with complex associated knowledge; they can grow to unmanageable size.

Rules

Consists of:

- a rule set for representing the expert knowledge
- a "database management system" for the case-specific facts
- a rule interpreter for problem solving

Example

IF: (1) stain of organism is Gram negative AND

(2) morphology of organism is rod AND

(3) aerobicity of organism is aerobic

THEN: strong evidence (0.8) that organism is Enterobact

Properties of rule-based systems:

- modularity of rule, very expressive, easy handling of certainty factors (probabilistic, possibilistic reasoning)
- Lack of precise semantics of rules. Not always efficient

Rules - A simple example

- Assume: Knowledge base consisting of facts and rules, a rule interpreter to match the rule conditions against facts and means for executing the rules.

Rules:

R1: IF: Raining,Outside(x),Has_Umbrella(x)
    THEN: Uses_Umbrella(x)

R2: IF: Raining,Outside(x)
        NOT Has_Umbrella(x)
    THEN: Wet(x)

R3: IF: Wet(x)
    THEN: Gets_Cold(x)

R4: IF: Sunny,Outside(x)
    THEN: Gets_Sun_Tan(x)

Initial facts: Raining, Outside(John)

Rules - A simple example

<u>Correct:</u>

- Only one rule, R2 matches the facts with [x→John], hence add Wet(John)
- Facts after first cycle:

  Raining, Outside(John), Wet(John)
- Now R3 matches facts, hence add

  Gets_Cold(John)
- Facts after second cycle:

  Raining, Outside(John), Wet(John), Gets_Cold(John)

<u>Incorrect:</u>

Gets_Sun_Tan(John)

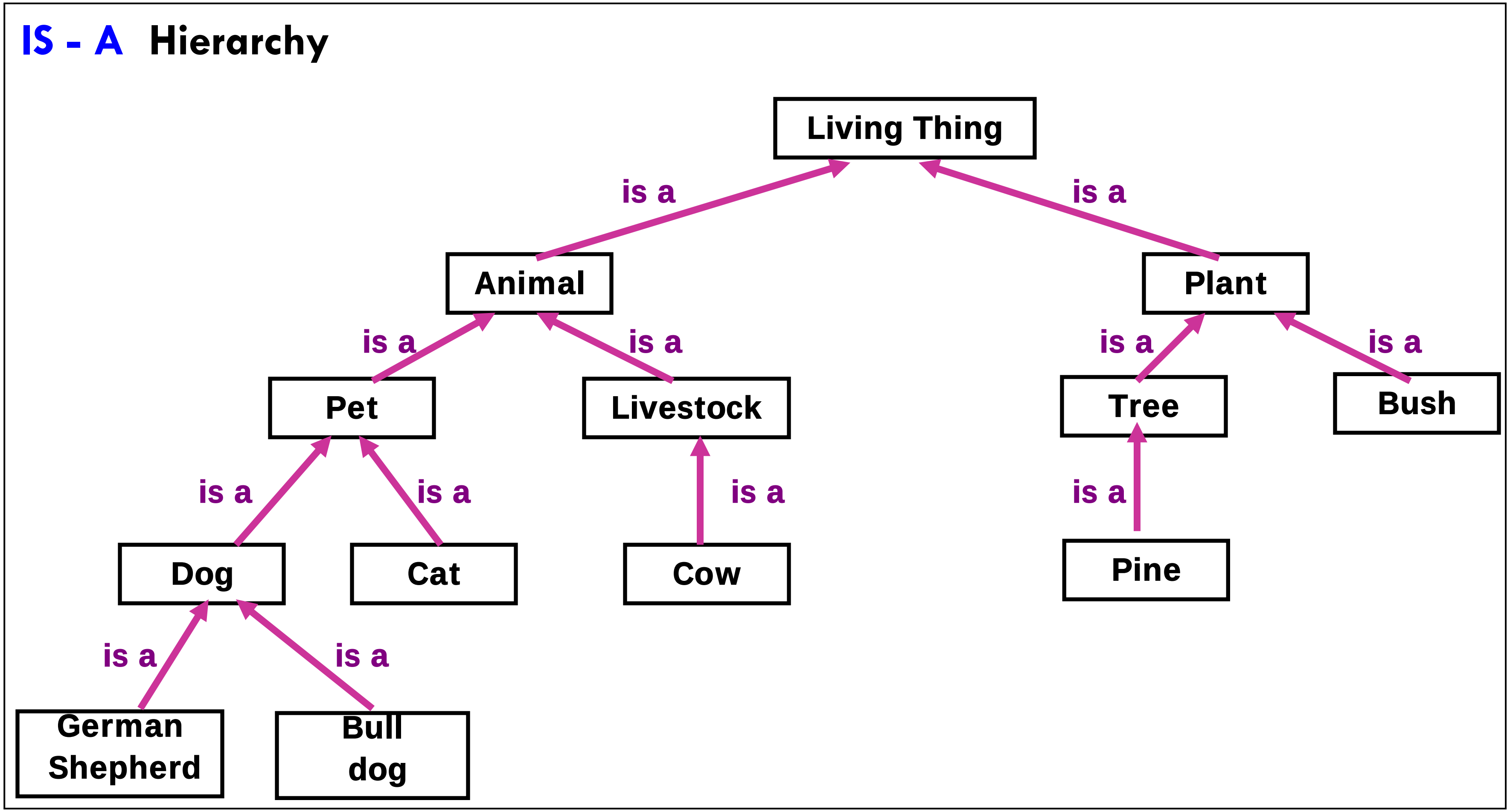
Process of deriving new facts from given facts, is called INFERENCE

Semantic Networks

- Semantic networks are graphical representation of entities and their relationships. The nodes are objects or events; the arcs are the relationships or moves.

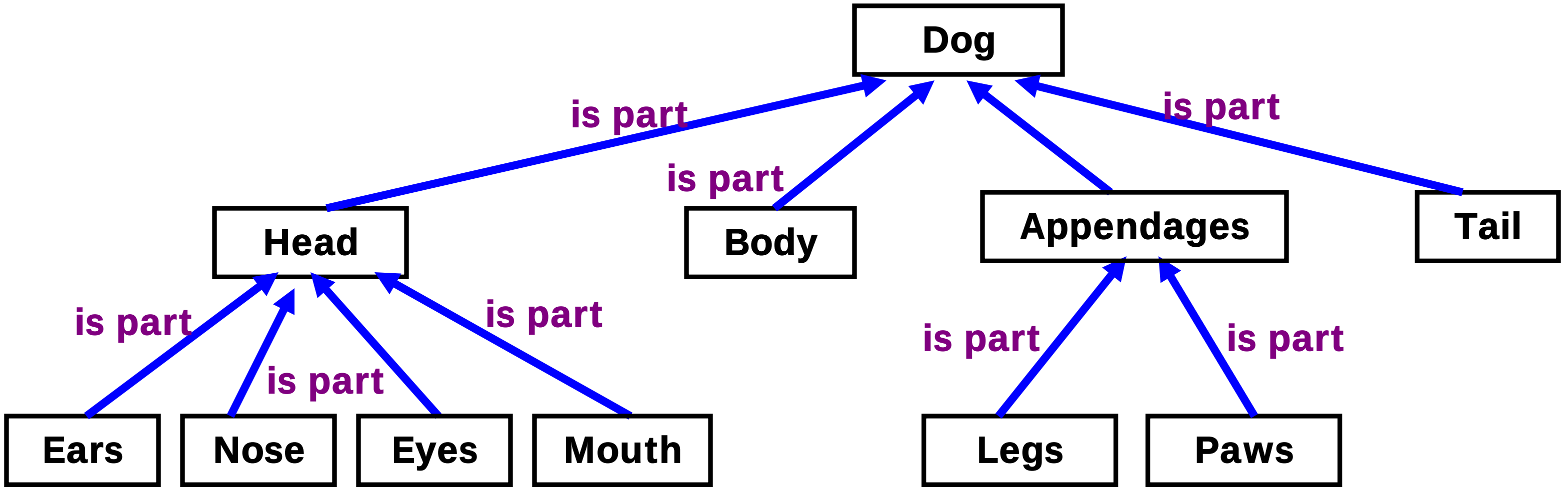
  - *Advantages*. Easy to translate to predicate calculus.
  - *Disadvantages*. Cannot handle quantifiers; nodes may have confusing roles or meanings; searching may lead to combinatorial explosion; cannot express standard logical connectives; can represent only binary or unary predicates.

Semantic Networks - An example 1

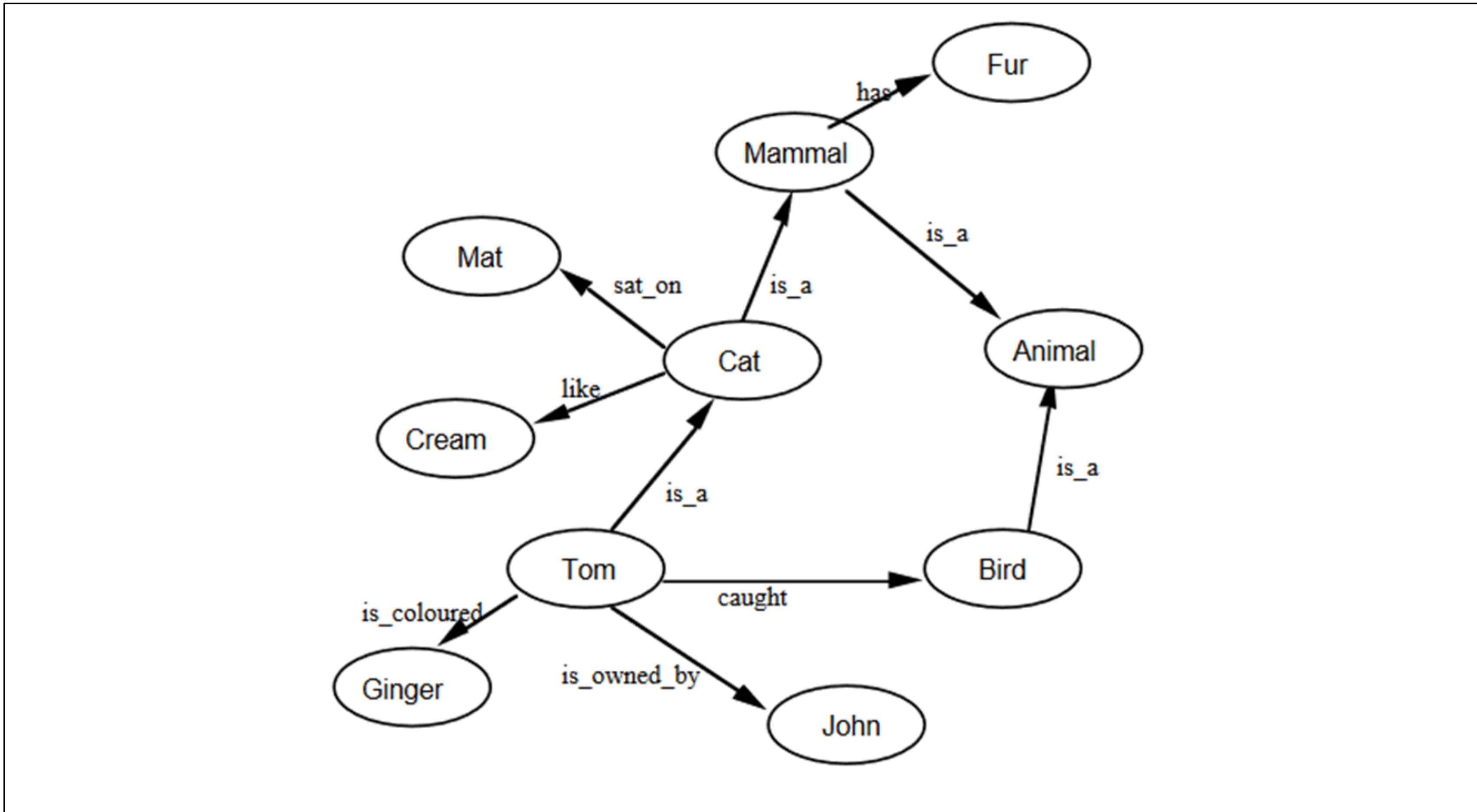


Semantic Networks - An example 2

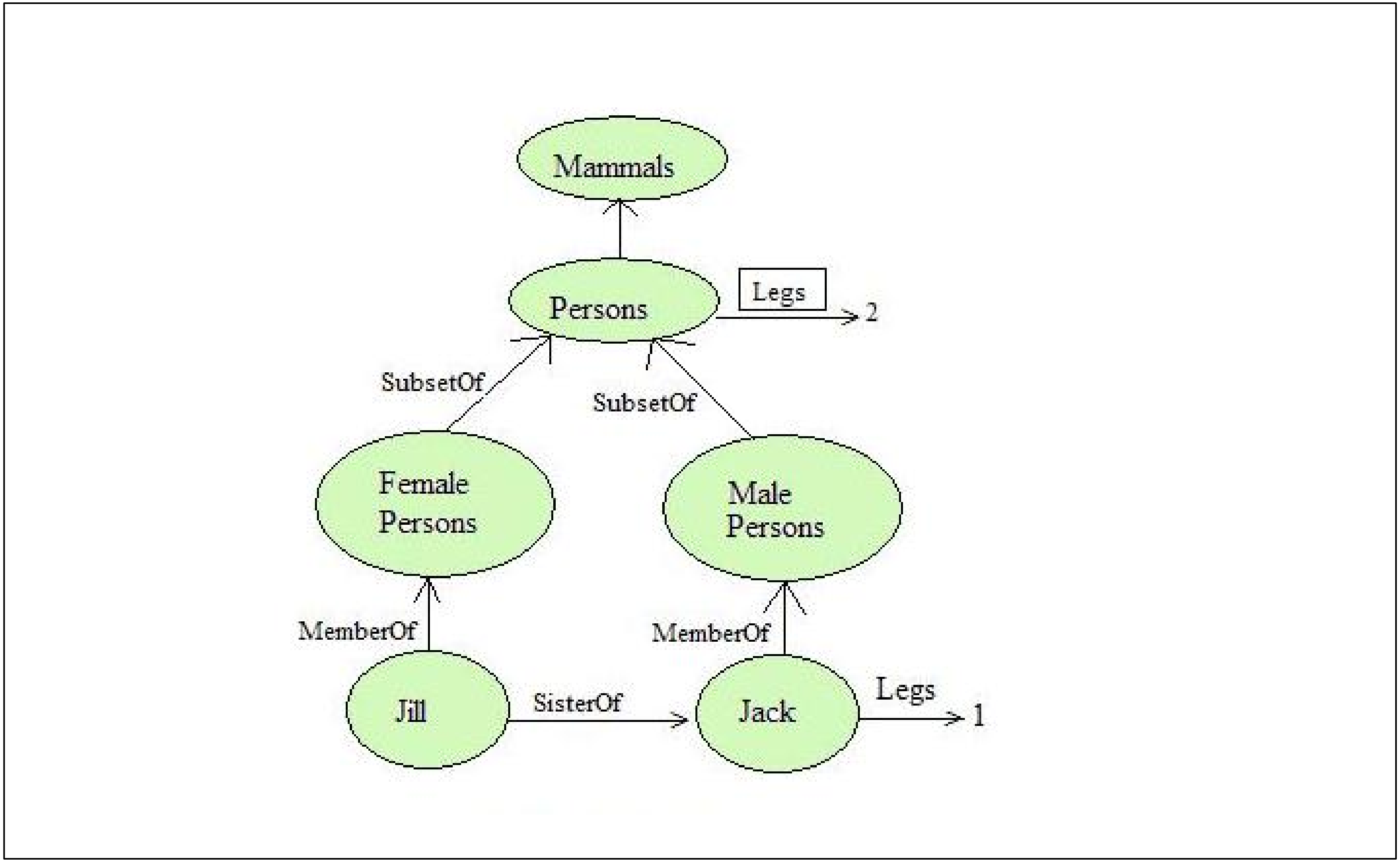
IS - PART Hierarchy



Semantic Networks - An example 3



Semantic Networks - An example 4



Propositional Logic

- **Propositional Logic (Propositional calculus/Boolean Logic)** - assertions describing things, use logical connectives and boolean logic

  - *Advantages*. Can reason about the world; based on proven theory
  - *Disadvantages*. Components cannot be individually examined

Propositional Logic

The syntax:

- Vocabulary:
  - A set of propositional symbols - e.g., P, Q, …
- A set of logical connectives or operators
  - usually $\vee$ (OR), $\wedge$ (AND), $\neg$ (NOT), $\rightarrow$ (implication), maybe $\Leftrightarrow$ (equivalence), Parenthesis (for grouping)

Read: Logic	$\vee$	$\wedge$	$\rightarrow$	$\neg$
Nat. Lang.	Or	And	Implies	Not

- The special symbols
  - True, False (logical constants)

Propositional Logic

Rules for forming sentences:

- Each symbol (i.e., a constant or a propositional symbol) is a sentence (an atomic sentence).
- A sentence in parentheses is a sentence.
- If P and Q are sentences, then so are
  - $P \vee Q$ (disjunction)
  - $P \wedge Q$ (conjunction)
  - $\neg P$ (negation)
  - $P \rightarrow Q$ (implication)
  - and similarly for whatever other connectives we allow

Propositional Logic

Sample sentences

- P
- True
- $P \vee Q$
- $\neg P$
- $(P \vee Q)$
- $\neg(P \vee Q)$
- $\neg P \vee Q$
- $(P \vee Q) \rightarrow R$
- $P \wedge \neg P$

What do the sentences mean?

- E.g., P might be "It is raining in Nyahururu", and Q, "Nairobi is a city"
- We interpret logical connectives in the obvious way.
  - E.g., $\neg P$ means that P is not the case; $P \vee Q$ means that at least one of P or Q is true.

Propositional Logic - example

- Let, Fact P: "Ali likes chips"
- Let, Fact Q: "Ali eats chips"

- Other possible facts:
  - $P \lor Q$: "Ali likes chips or Ali eats chips"
  - $P \land Q$: "Ali likes chips and Ali eats chips"
  - $\neg Q$: "Ali doesn't eat chips"
  - $P \rightarrow Q$: "If Ali likes chips then Ali eats chips"

Propositional Logic

Truth

- For sentences, we also get to say whether they are true or false.
  - True is always true; False always false.
  - P, Q, etc., are true or false depending on their interpretation.
    - So these are satisfiable, but not valid.
  - Complex sentences are true or false as a function of their connective.
    - Usually specified as a truth table.

Propositional Logic

Truth tables

Conjunction($\wedge$)

P	Q	$P \wedge Q$
false	false	false
false	true	false
true	false	false
true	true	true

Disjunction($\vee$)

P	Q	$P \vee Q$
false	false	false
false	true	true
true	false	true
true	true	true

Implication($\rightarrow$)

P	Q	$P \rightarrow Q$
false	false	true
false	true	true
true	false	false
true	true	true

Negation($\neg$)

P	$\neg P$
true	false
false	true

Equivalence($\leftrightarrow$)

P	Q	$P \leftrightarrow Q$
false	false	true
false	true	false
true	false	false
true	true	true

Propositional Logic

A Proof Theory for Propositional Logic

- It is easy to devise a procedure to determine the truth of an arbitrary sentence in propositional logic.
- Just write down a big truth table, and see if the sentence is always true.
- E.g., suppose we want to know if $\neg(P \wedge Q) \rightarrow \neg P \vee \neg Q$.
- Here is a truth table:

P	Q	$\neg(P \wedge Q)$	$\neg P \vee \neg Q$	$\neg(P \wedge Q) \rightarrow \neg P \vee \neg Q$
false	false	true	true	true
false	true	true	true	true
true	false	true	true	true
true	true	false	false	true

So we have proved this sentence.

Propositional Logic

Reasoning in Propositional Logic

- Similarly, if we assume a few things, we can determine if something follows.
- E.g., if we assume P, then $P \vee Q$, say, degenerates into $\text{True} \vee Q$, which a truth table will tell us is always true.
- So, we can always draw valid conclusions from premises, **regardless of what any of this means**

Propositional Logic

Reading Assignment - Common Inference Rules

- Modus Ponens: mode that affirms

  $\alpha \rightarrow \beta, \ \alpha \ |\!- \ \beta$

- And-Elimination: simplification

  $\alpha_1 \wedge \alpha_2 \ |\!- \ \alpha_i$

- And-Introduction: conjunction

  $\alpha_1, \alpha_2 \ |\!- \ \alpha_1 \wedge \alpha_2$

- Or-Introduction: addition

  $\alpha \ |\!- \ \alpha \vee \beta$

- Double-Negation Elimination:

  $\neg\neg \alpha \ |\!- \ \alpha$

- Resolution:

  $\alpha \vee \beta \, , \neg \beta \vee \gamma \ |\!- \ \alpha \vee \gamma$

Propositional Logic

- Propositional Logic has limitations, - it is not expressive enough
- First-Order Logic is an improvement and is useful

Predicate Logic (FOL)

- **Predicate Logic (Predicate Calculus / First Order Logic) -** is an extension of propositional Logic.
- **Predicates** that are used in FOL are of the form function(arguments), where function is any object or relationship and quantifiers are used.
- In FOL, the world consists of **objects**, i.e. things with individual identities and **properties** that distinguish them from other objects.
- Among these objects, various **relations** hold. Some of these relations are **functions**. i.e. relations in which there is only one "value" for a given "input".

First-Order Logic (FOL)

Examples:

- Objects: people, houses, colours, the moon, Mutua,...
- Relation: brother of, bigger than, inside, part of, owns,..
- Properties: red, round, healthy, tall..
- Functions: father of, best friend,...

- *Advantages*. It has well defined rules for manipulation; it is expressive.
- *Disadvantages*. Cannot handle uncertainty; uses small primitives for descriptions whose numbers can be many.

First-Order Logic

Read: Logic	$\vee$	$\wedge$	$\rightarrow$	$\neg$	$\forall$	$\exists$
Nat. Lang.	Or	And	Implies	Not	Forall	Exists

Examples: (a) man(Pat)

(b) married(Pat,Jan)

(c) $\forall x \forall y$[[married(x,y) $\wedge$ man(x)]$\rightarrow \neg$man(y)]

(d) man(Jan) $\vee$ woman(Jan)

(e) $\forall x \exists y$[person(x)$\rightarrow$has_mother(x,y)]

Properties of First-Order Logic:

- very expressive as well as unambiguous syntax and semantics
- no generally efficient procedure for processing knowledge

Correct:

raining
raining$\rightarrow$street_wet
street_wet

Incorrect:

raining
raining$\rightarrow$street_wet
elephant_hungry

First-Order Logic

Syntax:

- Connectives - $\Rightarrow$, $\wedge$, $\vee$, $\Leftrightarrow$
- Quantifiers - $\forall$, $\exists$
- Constants - A, X_1,,Mutua, Alice,..
- Variables - a, x, s,..
- Predicate - Before, HasColour, Raining,..
- Functions - Mother, LegOf,..

First-Order Logic

Connections between $\forall$ and $\exists$

- e.g. "Everybody likes ice cream" means that there is no one who doesn't like ice cream.
  - $\forall x\ likes(x,icecream) \Leftrightarrow \neg\exists x\ \neg likes(x,icecream)$

- Using De Morgan's Theorem:
- $\forall x\ S \Leftrightarrow \neg\exists x\ \neg S$
- $\exists x\ S \Leftrightarrow \neg\forall x\ \neg S$

Examples: Translate into Logic

- Some dogs bark

  $\exists$ x. Dog x $\wedge$ Barks x

  $\exists$ xy . Dog x $\wedge$ Bark y $\wedge$ makes_sound x

- All dogs have four legs

  $\forall$ x . Dog x $\rightarrow$ $\exists$ y z w u . Leg y $\wedge$ Leg z $\wedge$ Leg w $\wedge$ Leg u $\wedge$ has x y $\wedge$ has x z $\wedge$ has x w $\wedge$ has x u $\wedge$ y $\neq$ z $\wedge$ y $\neq$ w $\wedge$ y $\neq$ u $\wedge$ z $\neq$ w $\wedge$ z $\neq$ u $\wedge$ w $\neq$ u

- All barking dogs are irritating.

  $\forall$ x . Dog x $\wedge$ Barking x $\rightarrow$ Irritating x

Examples: Translate into Logic

- No dogs purr.

  $\neg \exists x . \text{Dog } x \wedge \text{Purrs } x$

  $\forall x . \text{Dog } x \rightarrow \neg \text{Purrs } x$

- Fathers are male parents with children

  $\forall x . \text{Male } x \wedge \text{Parent } x \wedge (\exists y . \text{Child } y \text{ and has } x\ y) \leftrightarrow \text{Father } x$

- Students are people who are enrolled in courses.

  $\forall x . \text{Student } x \leftrightarrow \text{Person } x \wedge (\exists y . \text{Course } y \wedge \text{enrolled\_on } x\ y)$

Frames

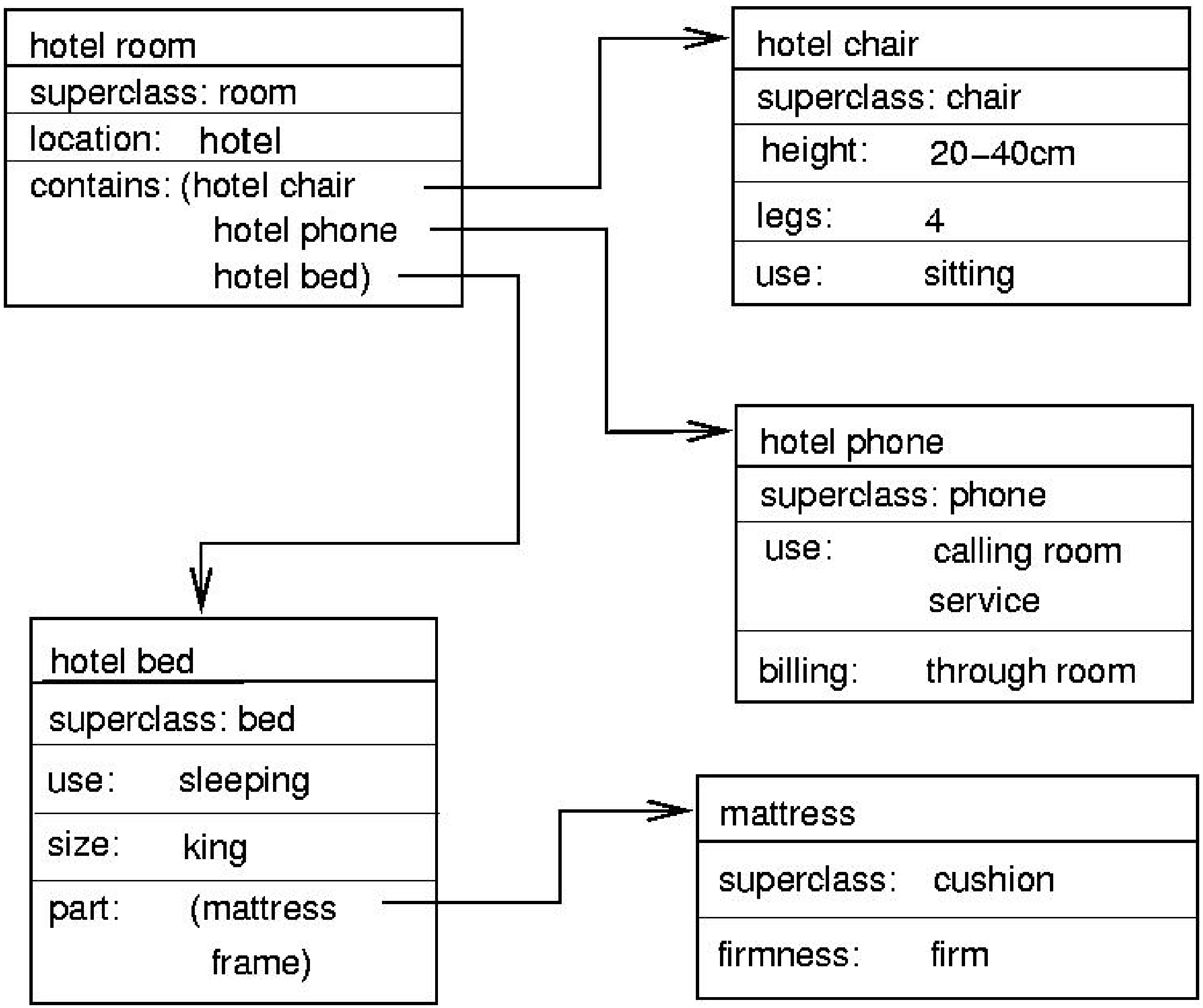
- A frame is an AI data structure used to divide knowledge into substructures which represent “stereotyped situations.”
- Frames are also useful for representing commonsense knowledge
- A frame represents an object that is typical to a stereotypical situation
- Objects are arranged in a hierarchical manner
- Frames can be derived from semantic nets
- A frame comprises of slots and slot filler

Frames - example 1

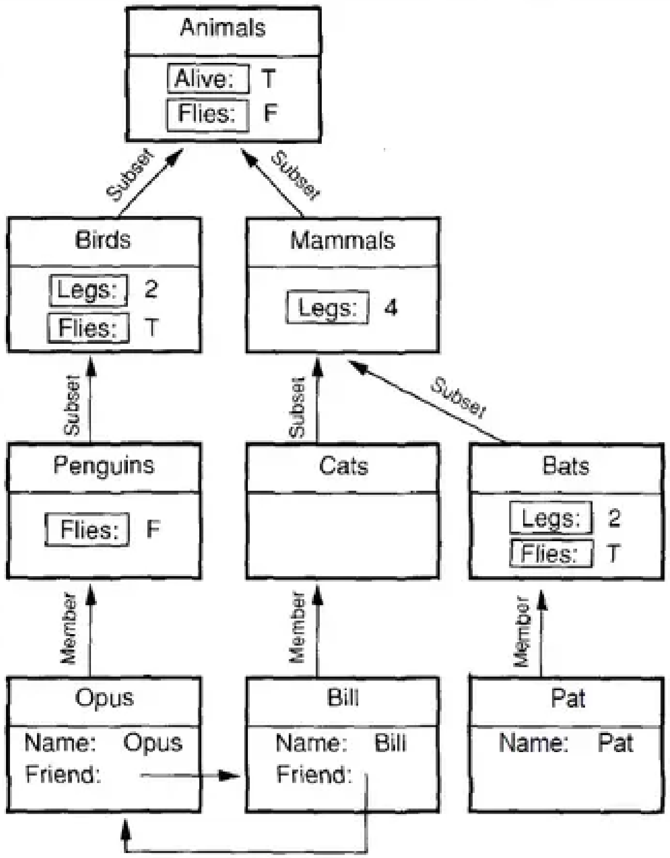
Slots	**Fillers**
publisher	Thomson
title	Expert Systems
author	Giarratano
edition	Third
year	1998
pages	600

Frame Name	Vacation
Where	Albury
When	March
Cost	$1000

Frames - example 2



Frames - example 3



(a) A frame-based knowledge base

Rel(Alive,Animals,T)
Rel(Flies,Animals,F)

Birds $\subset$ Animals
Mammals $\subset$ Animals

Rel(Flies,Birds,T)
Rel(Legs,Birds,2)
Rel(Legs,Mammals,4)

Penguins $\subset$ Birds
Cats $\subset$ Mammals
Bats $\subset$ Mammals
Rel(Flies,Penguins,F)
Rel(Legs,Bats,2)
Rel(Flies,Bats,T)

Opus $\in$ Penguins
Bill $\in$ Cats
Pat $\in$ Bats
Name(Opus,"Opus")
Name(Bill,"Bill")
Friend(Opus,Bill)
Friend(Bill,Opus)
Name(Pat,"Pat")

(b) Translation into first-order logic

Figure 10.7 A frame-based network and a translation of the network into first-order logic. Boxed relation names in the network correspond to relations holding for all members of the set of objects.

Representation of Knowledge - Summary

- There is no single most adequate knowledge representation formalism/scheme for everything.
- Main points for selecting a representation formalism: what should be represented, how should the knowledge be processed.
- There are many more representation formalisms. All the above mentioned are symbolic. There are non-symbolic (e.g. pictorial) ones. Neural networks work on non-symbolic representations.